**MODELACIÓN DE OBJETOS**

Primero que todo hay que tener en cuenta los elementos del juego: mira de la resortera, mangos, animales ladrones, animales tranquilos, árbol.

**LISTA DE OBJETOS Y DECLARACIÓN (*clases*).** Todos los siguientes objetos incluirán variables que definan su posición en la escena y también variables fijas que definan su tamaño:

1. ***mira****.* Esta clase solo será de guía para la clase que contendrá los *sprites* de la piedra, por lo que muchas de sus variables deben ir en un entorno público para un manejo más fácil de las variables.
2. ***piedra.*** Esta clase manejará *sprites* de una piedra que permitan simular el efecto de profundidad. La posición de la piedra siempre será la misma y estará dada por el objeto ‘*mira*’[[1]](#footnote-1).
3. ***mango***. Debido a que por ahora no sé que variables debe tener un objeto al que se le van a aplicar físicas, la definición puede ser en un inicio muy vacía. Básicamente necesitará funciones que le permitan moverse cuando esté aplicando las físicas y/o cuando esté siendo transportada por un animal ladrón.
4. ***animal****.* Funcionará como una clase padre que contendrá todas las especificaciones que las clases ‘*ladron*’ y ‘*quiet*’ necesitarán y tendrán en común. Por ejemplo, manejo de *sprites*, propiedades físicas y propiedades de visibilidad. En cuanto a la visibilidad, aparecerá con un efecto de barrido de abajo hacia arriba y desaparecerá después de cierto tiempo (se maneja con la clase *QTimer*) con un efecto que simule el esconderse entre las hojas del árbol.
5. ***ladron****.* Esta clase heredará de manera pública a la clase ‘*animal*’. Además, se le agregará funciones de movilidad donde se incluyan las físicas que utilizará (M.R.U., parabólica, caída libre, elasticidad).
6. ***quiet***. Esta clase heredará de igual manera como lo hace ‘*ladron*’, pero en esta clase por ahora no se necesita agregar o redefinir métodos.
7. ***ramas.*** Servirán como plataformas para el recorrido de los animales ladrones, por lo que no tendrán funcionalidades de moverse, aparecer o desaparecer.

**RESULTADO DE LA DECLARACIÓN DE OBJETOS:**

class mira : public QGraphicsItem {

*public*:

int posx, posy, ancho, alto;

Se inicializa definiendo los valores fijos de las variables anteriores

mira();

QRectF boundingRect() const;

void paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

Demás variables, métodos, *signals* y *public slots* necesarios para el seguimiento con el *mouse*.

………………

};

class piedra : public QObject, public QGraphicsItem {

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

*private*:

int posx, posy, radio;

*public*:

piedra();

piedra(int x, int y, int r);

QRectF boundingRect() const;

QTimer \*timer;

QPixmap \*pixmap;

void paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

int velocidad;

float fila, columnas; ***manejo de sprites***

signals:

public slots:

void Actualizacion();

};

class mango : public QGraphicsItem

*public*:

int posx, posy, ancho, alto;

mango();

Variables necesarias para el movimiento pendular (masa, cuerda, punto fijo, velocidad, periodo)

QRectF boundingRect() const;

……………

Movimiento natural del objeto durante todo el juego.

moverPendulo();

moverParabola();

moverLinealArriba();

Movimientos que adquiere cuando es robado por un animal.

moverLinealAbajo();

moverLinealIzquierda();

moverLinealDerecha();

class animal : public QObject, public QGraphicsItem {

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

*private*:

int posx, posy, ancho, alto;

*public:*

animal();

/\*Omito todos los métodos para sprites y las variables que hacen falta para los siguientes métodos\*/

Barrido de abajo hacia arriba

mostrarAnimal();

Efecto para esconder

esconderAnimal();

caerLibre();

Física de caída libre

};

class ladron : public animal {

Q\_OBJECT

Q\_INTERFACES(QGraphicsItem)

*private*:

int posx, posy, ancho, alto;

*public:*

ladron();

/\*Omito todos los métodos para sprites y las variables que hacen falta para los siguientes métodos\*/

MRU hacia la izquierda

correrRight();

correrLeft();

MRU hacia la derecha

saltar();

Choque elástico

Movimiento parabólico

choque();

};

Por ahora no se define la clase quiet ya que sería la misma que ‘animales’.

Las ramas no necesitan que se les cree una clase aparte ya que los objetos de las librerías de Qt los contiene y los soporta.

1. La mira y la piedra simulada deben ir en clases diferentes debido a que en el manejo de los *sprites* el tamaño del objeto debe ser el mismo que el de todas las imágenes, pero el tamaño del objeto mira es considerablemente menor que el tamaño de los primeros *sprites* de la piedra. [↑](#footnote-ref-1)